General Disclaimer

One or more of the Following Statements may affect this Document

- This document has been reproduced from the best copy furnished by the organizational source. It is being released in the interest of making available as much information as possible.

- This document may contain data, which exceeds the sheet parameters. It was furnished in this condition by the organizational source and is the best copy available.

- This document may contain tone-on-tone or color graphs, charts and/or pictures, which have been reproduced in black and white.

- This document is paginated as submitted by the original source.

- Portions of this document are not fully legible due to the historical nature of some of the material. However, it is the best reproduction available from the original submission.

NASA TECHNICAL MEMORANDUM

NASA TM -73869

NASA TM-73869

AN AUTOMATED PROCEDURE FOR CALCULATING SYSTEM MATRICES FROM PERTURBATION DATA GENERATED BY AN EAI PACER 100 HYBRID COMPUTER SYSTEM

by Edward J. Milner and Susan M. Krosel
Lewis Research Center
Cleveland, Ohio 44135

December 1977

1. Report No. NASA TM-73869	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle AN AUTOMATED PROCEDURE FOR CALCULATING SYSTEM MATRICES FROM PERTURBATION DATA GENERATED BY AN EAI PACER 100 HYBRID COMPUTER SYSTEM		5. Report Date 6. Performing Organization Code
7. Author(s) Edward J. Milner and Susan M. Krosel		8. Performing Organization Report No. E-9465 10. Work Unit No.
9. Performing Organization Name and Address National Aeronautics and Space Administration Lewis Research Center Cleveland, Ohio 44135		11. Contract or Grant No. 13. Type of Report and Period Covered Technical Memorandum
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D. C. 20546		14. Sponsoring Agency Code
15. Supplementary Notes		

16. Abstract

Techniques are presented for determining the elements of the A, B, C, and D state variable matrices for systems simulated on an EAI Pacer 100 hybrid computer. An automated procedure systematically generates disturbance data necessary to linearize the simulation model and stores these data on a floppy disk. A separate digital program verifies this data, calculates the elements of the system matrices, and prints these matrices appropriately labeled. The partial derivatives forming the elements of the state variable matrices are approximated by finite difference calculations.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Hybrid simulation State variables Linear modeling	Unclassified - unlimited STAR Category 61

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified		

* For sale by the National Technical Information Service, Springfield, Virginia 22161

AN AUTOMATED PROCEDURE FOR CALCULATING SYSTEM MATRICES FROM PERTURBATION DATA GENERATED BY AN EAI PACER 100 HYBRID COMPUTER SYSTEM

by Edward J. Milner and Susan M. Krosel

National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135

E-9465

SUMMARY

Most control system design techniques assume having available a linear representation of the system to be controlled. Although obtaining a linearized model of a complex, non-linear system is usually tedious and time consuming, this task can be made less so by using the automated procedure described in this report.

Techniques are presented for determining the elements of the A, B, C, and D state variable matrices for systems simulated on an EAI Pacer 100 hybrid computer system. The techniques are generalized to allow use with many different simulation models. An automated procedure systematically generates disturbance data necessary to linearize the simulation model and stores these data on an auxiliary TEKTRONIX model 4922 floppy disk.

A separate digital program verifies the floppy disk data, allows for data modification, if necessary, calculates approximations to the partial derivatives forming the elements of the system matrices, and prints the matrices appropriately labeled.

Use of the floppy disk as a storage medium permits the data generation process on the hybrid computer to be separate from the matrix element calculation process.

INTRODUCTION

The intricate nature of many dynamic systems necessarily requires rather complex simulation models to represent performance over their entire operating range. One is likely to find the model to include a combination of linear and nonlinear algebraic and differential equations and maps. Although such a model is useful for predicting system performance over a wide range of operating conditions, generally, its nonlinear characteristics hinder it from being well-suited for control analyses. A basic assumption of most control design techniques is having available a linear representation of the system to be controlled.

By considering operation in the vicinity of a particular operating point, state variable techniques can be employed to obtain a linear approximation of the nonlinear system. Doing so, the system equations can be written in vector notation as:

$$\dot{x} = Ax + Bu \tag{1}$$

$$y = Cx + Du \tag{2}$$

where x is a vector of the system time-varying states, $\dot{x}$ is a vector of the time rate of change of the system states, u is a vector of system inputs, and y is a vector of system outputs. Matrices A, B, C, and D are characterized by:

$$a_{ij} = \frac{\partial \dot{x}_i}{\partial x_j} \tag{3}$$

$$b_{ij} = \frac{\partial \dot{x}_i}{\partial u_j} \tag{4}$$

$$c_{ij} = \frac{\partial y_i}{\partial x_j} \tag{5}$$

$$d_{ij} = \frac{\partial y_i}{\partial u_j} \tag{6}$$

One linearization technique is to obtain finite difference approximations for the partial derivatives which comprise the elements of the system matrices. To generate the partial derivatives, the steady-state level of each state variable and system input is independently stepped in both a positive and negative direction while holding the other state variables and system inputs fixed. Corresponding values of the state variable derivatives and system outputs are recorded for each step change. By calculating the deviation of each state variable derivative and output from its steady-state value and by forming appropriate quotients the matrix elements are formed.

This report describes an automated technique for determining the elements of the A, B, C, and D state variable matrices for systems that are simulated on an EAI Pacer 100 hybrid computer system. The technique is generalized to allow its use with many different simulation models. The data taking process is initiated by manually setting a sense switch at the Pacer 100 control console. The order in which the variables are to be perturbed along with other appropriate data are input from cards. Parameter values are transmitted from

the analog computer to the digital computer by means of analog-to-digital converters (ADC's). Perturbation data generated on the Pacer 100 system are recorded and stored on a TEKTRONIX model 4922 floppy disk. The data are later transferred from the floppy disk to an IBM TSS/360 computer where the matrix calculations are performed. The matrix calculation is contained in the program MATCALC which can generate matrices for systems having an arbitrary maximum of 20 states, 10 inputs, and 20 outputs.

The following sections provide details of the apparatus, the digital program, the perturbation process, the floppy disk data format, and the matrix calculation procedure. An example is provided to illustrate the procedure.

APPARATUS

The data taking technique, described in this section, applies to nonlinear system simulations implemented on an EAI Pacer 100 hybrid computer system. The technique could, of course, be modified for other hybrid computer systems. At the NASA Lewis Research Center, the Pacer 100 system includes the Pacer digital computer with 32K of core, a TEKTRONIX model 4010 terminal with auxiliary TEKTRONIX model 4922 floppy disk unit, two model 681 analog computers, and an interface to allow communication between the computers. That interface consists of 48 analog-to-digital converters (ADC's) and 24 digital-to-analog converters (DAC's).

Analog Setup Requirements

It is assumed that the integrations associated with the system dynamics are performed by the integrators on one or both of the analog computers. The integrator outputs then correspond to the system state variables. The perturbation program requires that each state variable, state derivative, input variable, and output variable be transmitted from the analog computer to the digital computer through the interface. Hence, each parameter of interest must be connected to an ADC. For the purposes of this report, parameters formulated on the "primary" analog console are transmitted to the digital computer via a block of ADC's beginning with ADC 1 and parameters formulated on the "secondary" analog console are transmitted via a block of ADC's beginning with ADC 33.

Digital Program Description

Appendix A contains FORTRAN listings of the major parts of a typical simulation program which incorporates the necessary perturbation logic.

The program MAIN is designed to control the execution of the major parts of the program. As the execution of each part is

completed, control is returned to MAIN. The analog computer setup and checkout takes place during the execution of subroutine SET. This includes setting and checking analog pots, digitally controlled attenuators (DCA's), and digitally controlled function generators (DCFG's). Subroutine STATIC provides for the reading of all amplifier outputs at any desired operating point. Subroutine LOOP is the heart of the simulation insofar as this is the area where the dynamics of the system being simulated are updated. By continually cycling through component maps and table lookups, current values of the system state derivatives are obtained. In general, LOOP will receive information from the analog computer and output information to the analog computer through the interface.

The perturbation program consists of three primary digital subroutines - DATOUT, SGNFIX, and AVG. The function served by subroutine DATOUT is to set the integrator initial condition DCA's to correspond to the steady-state operating point being considered and to systematically perturb the state variables and the system inputs one at a time in both a positive and negative sense. DCA's are used because they can be set by the digital computer without changing modes on the analog computer. Subroutine SGNFIX is used to guarantee that the positive algebraic sign of each parameter is recorded on floppy disk. For example, consider a variable, x, whose range spans the set of all real numbers. Hence, x may have both positive and negative real values. However, the analog computer representation can be either the variable (+x) or the variable (-x) depending on the sign of the reference voltage used. If (-x) were the analog computer variable, subroutine SGNFIX converts it to (+x). For each perturbation, subroutine AVG scans the parameters of interest forming 100 point averages for each of them. These data are then recorded on floppy disk.

Formatting the output data and recording it on floppy disk is accomplished in subroutines ADDUP and DISK, respectively. The sole purpose of subroutine DISK is to control the recording of the data on floppy disk. To record on floppy disk, the floppy disk unit is plugged into the disk interface card on the TEKTRONIX 4010 CRT terminal. The write statements in the program are the same as if the data were being written on the CRT terminal. Data is then automatically written on the CRT terminal and recorded on the floppy disk. Note that everything written on the CRT terminal is also recorded on the floppy disk.

Illustrative Example

Consider the following system of four uncoupled dynamic systems linked through output relationships:

$$ST81 = \int(INPUT1-ST81)dt \quad \text{where } |INPUT1| \leq 2 \text{ and } |ST81| \leq 2 \tag{7}$$

$$ST82 = \int(INPUT2-ST82)dt \quad \text{where } |INPUT2| \leq 2.5 \text{ and } |ST82| \leq 2.5 \tag{8}$$

$$ST83 = \int(INPUT3-ST83)dt \quad \text{where } |INPUT3| \leq 5 \text{ and } |ST83| \leq 5 \tag{9}$$

$$ST84 = \int(INPUT4-ST84)dt \quad \text{where } |INPUT4| \leq 4 \text{ and } |ST84| \leq 4 \tag{10}$$

An analog diagram representing a typical system such as these is presented in Figure 1. Equations (7) and (8) are to be programmed on analog console 1. DCA 45 is to be used as the initial condition pot associated with ST81 and DCA 41 is to be used to set the perturbation amplitude of INPUT1. Likewise, DCA's 75 and 71 shall be used with ST82 and INPUT2, respectively.

Correspondingly, Eqs. (9) and (10) are to be programmed on analog console 2. However, on this console DCA's 75, 71, 45, and 41 shall be used with ST83, INPUT3, ST84, and INPUT4, respectively.

In all cases a 2 percent perturbation amplitude is deemed sufficient.

The solution of this example problem will be followed through in the discussion which follows.

PROCEDURE

The data taking process is activated by setting sense switch 4 and then exiting from the main system calculation subroutine (called LOOP) by setting sense switch 1. Control is then transferred to DATOUT. Hereafter, DATOUT controls reentry into LOOP. An internal check is made that the analog computer was in the operate mode at the time the data taking process was activated. Referring to Figure 2, a card is then read in (3I3) format giving the number of state variables, the number of system inputs, and the number of system outputs formulated on the primary analog console only. A series of cards punched in (F5.1, I3, S7) format are then read. Each card contains information characterizing one of the parameters of interest. For each state variable, the algebraic sign of the integrator output, the initial condition pot number, and the proportion of perturbation amplitude desired are required. For each system input, its algebraic sign, its perturbation pot number, and the proportion of perturbation amplitude desired are required. For each system output only its algebraic sign (as previously discussed in the explanation of subroutine SGNFIX) is required.

The next card read is a card in (3I3) format giving the number of state variables, the number of system inputs, and the number of system outputs formulated on the secondary analog console. Similarily, another series of cards in (F5.1, I3, S7) format are then read characterizing the parameters of interest on the secondary analog console as was just done for the primary console. Table I contains a listing of the cards used for the example and Figure 2 shows the data deck setup.

The order in which these data cards are read is very important. The parameter order determined by the cards must correspond with the parameter order determined by the analog ADC's used. State variable data is followed by system input data, which, in turn, is followed by system output data. Also, the order in which the parameters appear within each of the first two groups is important insofar as the order in which the cards are read determines the order in which the perturbations will occur. Note that the order of parameter perturbations is in itself arbitrary. Limitations on this order are imposed only by the analog console on which the integrator for a state variable or system input is located. As this procedure is defined, all state variables and system inputs on the primary analog console will be perturbed first. Then those on the second analog console will be perturbed. Table II gives the order in which the state variables and system inputs of the example are perturbed. Also, given is the analog console for each parameter and its associated ADC channel.

Following the reading of these data cards, the digital program proceeds to monitor the steady-state integrator outputs and sets DCA's corresponding to the integrator initial condition values. After the steady-state operating point is established, the analog computer is automatically put in IC mode where it remains during the perturbation and data taking process. The data taking process begins with a baseline reading of the steady-state values of the parameters of interest and the recording of them on floppy disk. The set of baseline data is designated as reading 1. Following the taking of this baseline data, each state variable and system input is stepped sequentially in both a positive and negative sense. Table III identifies the perturbations associated with each reading for the example.

For each perturbation, the data written on floppy disk are averages of the parameters of interest. The ADC's are read sequentially 100 times with the corresponding values summed and averaged. Since a parameter may be represented on the analog computer with either positive or negative algebraic sign, each average is multiplied by either (+1) or (-1), depending on the parameter sign. Thus, the data recorded on floppy disk corresponds to the positive algebraic sign of each parameter.

Data associated with each perturbation is recorded on floppy disk and each set of data on floppy disk is assigned its own identifying

reading number. Reading numbers are incremented as each set of data is recorded on floppy disk. Data is recorded on floppy disk according to the following format. The reading number is recorded twice in (2I5) format. The reason for this redundancy is so that it may be used as a check when reading the floppy disk. The parameter averages are recorded on floppy disk as integers (with 32767_{10} representing full scale on the 32K Pacer) at the rate of 18 values per line together with a line checksum according to the format (18I6,I7). The checksum value is the sum of all the integers output on that particular line. If the number of averages exceeds 18 but is not evenly divisible by 18, the second last line of output will be the n remaining averages recorded in (nI6) format and the final line will be the corresponding checksum in (I7) format. If the number of averages is less than 18, the n averages will be recorded in (nI6) format with a second line containing the checksum in (I7) format. An example of the floppy disk data format is given in Table IV for the test case.

The final characters required on each line are a "line feed" and "X-off". This ending sequence is necessary to insure proper data transfer to the IBM TSS/360 computer. Because this is not the normal line termination, the contents of the digital computer core location controlling line termination must be changed. The address of this core location is obtained by taking the address of .FOR from the memory map (for the listing given in Appendix A, this address is 172348) and adding 213_8. The address displacement value, 213_8, depends on the version of the .FOR routine used. Version E01, December 1975 is in use on the Pacer 100 system at Lewis. After loading the program, this memory cell is opened and its contents set to zero by the user.

The entire procedure is repeated for each perturbation. When the data taking process is completed, an appropriate message is output on the line printer. At this point the system may be returned to normal operation by resetting sense switches 1 and 4.

MATRIX CALCULATION TECHNIQUE

This section describes a matrix calculation technique which uses the IBM TSS/360 computer with data input provided by the floppy disk. The matrix calculation is contained in the program MATCALC. This is a generalized version of a program developed at Lewis for matrix generation from data obtained from a hybrid simulation of the F100 engine.

The MATCALC program has three basic sections: data input, data verification, and matrix generation. The data input section uses a software checksum to insure that the floppy disk data was correctly transmitted to the IBM TSS/360 computer. After the data has been correctly transmitted, the data input section unscales it and stores this unscaled data in an internal program matrix for use in the other two program sections. The data verification section determines: (1) if only one state or input was perturbed for each reading and (2) if the

perturbation size of each state or input was at least 1% of the state's or input's steady-state value. The matrix generation section calculates the approximations to the partial derivatives forming the elements of the A, B, C, and D matrices and prints out the matrices in their final form. A listing of the MATCALC program is given in Appendix B.

Transfer of data from the floppy disk to the 360 computer is controlled by a procdef called READIN. A listing of this procdef is included in Appendix B and the procedure for use is given in Appendix D. The procdef, READIN creates the vs-dataset, 'DTRK' which contains the floppy disk data. The data on disk is a series of readings where each reading is a list of values of the states, state derivatives, inputs, and outputs. The required format of each reading was discussed in the preceding section. There should be $2n+1$ readings on disk for each set of matrices, where n is equal to the number of states plus the number of inputs. The first reading is the 'base run' which is a reading of the unperturbed, steady-state values of the states, etc. (NOTE: The MATCALC program terminates if there is no 'base run'.) Corresponding to each reading is a disturbance number. The 'base run' must be assigned the disturbance number of 1. Since each state and input is perturbed in both a plus and minus direction, there are two readings associated with each state and input. These readings should be assigned disturbance numbers of m and $m+1$ where m is an even integer in the range 2 to $2n$.

The readings contained in dataset, 'DTRK', are used by the data input section of MATCALC to form an internal program matrix, DATA. The rows of the DATA matrix are the readings in the same order as they appeared in the floppy disk data and in dataset 'DTRK'. Each row contains the unscaled values of the corresponding reading (i.e., the nth row contains the unscaled values in the nth reading). The order of the values is the same for the reading and the row. Table V lists the DATA matrix for the example.

The rows of the DATA matrix are used in the data verification and the matrix generation sections of MATCALC. The INDX array, also formed in the data input section, is used in referencing individual rows of the DATA matrix. Table V also includes the INDX array for the example. This referencing is accomplished by:

Row No. of DATA matrix = INDX (DISTURBANCE NO.)

If the Row No. is zero, the reading data associated with that disturbance number was not contained in the floppy disk data (i.e., the data for a plus or minus perturbation of a particular state or input was not recorded).

The NAME array associates each state or input with two disturbance numbers: one for the plus perturbation and the other for the minus

perturbation. This array contains the names of the state variables and the system inputs in the order in which they are perturbed (see Table II). The data for the NAME array is contained in the input dataset, 'QINFO'. 'QINFO' is a vs-dataset created by the user and contains data describing the contents of the floppy disk and specifying the structure of the A, B, C, and D matrices. The format of 'QINFO' is given in Appendix C and Table VI lists the 'QINFO' dataset for the example.

Appendix D contains a detailed procedure for processing of perturbation data which has been stored on a floppy disk. Instructions for operating the TEKTRONIX 4010 CRT terminal and TEKTRONIX 4922 floppy disk drive unit are included. IBM TSS/360 commands for reading the floppy disk data into the IBM TSS/360 computer and for performing the matrix calculations are provided. Also in Appendix D is the procedure to follow should checksum errors occur.

Table VII shows the resulting matrix listing for the example. As the results show, the simulated system consisted of four uncoupled dynamic systems linked only through the output relationships. The linearized system equations are:

$$\dot{x}_1 = -0.9938\ x_1 + 0.9922\ u_1 \tag{11}$$

$$\dot{x}_2 = -0.9918\ x_2 + 0.9888\ u_2 \tag{12}$$

$$\dot{x}_3 = -0.9939\ x_3 + 0.9878\ u_3 \tag{13}$$

$$\dot{x}_4 = -0.9923\ x_4 + 0.9861\ u_4 \tag{14}$$

$$y_1 = 1.003\ x_1 - 0.001022\ x_3 \tag{15}$$

$$y_2 = -0.004666\ x_1 + 1.002\ x_2 + 0.001022\ x_3 + 0.001019\ u_3 \tag{16}$$

$$y_3 = 0.999\ x_3 - 0.001546\ x_4 \tag{17}$$

$$y_4 = 1.002\ x_4 \tag{18}$$

Program Considerations

Some modification of the perturbation program would be necessary if one of the state variables or inputs were at a limiting value while perturbation data were being taken. Because of that limit, that parameter could not be perturbed in both a positive and negative sense. Hence, proper logic would have to be included in subroutine DATOUT to restrict the perturbation of that parameter to the direction which would not violate the limiting value. Likewise, if the problem size were such that only one analog console were required, subroutine DATOUT could be easily modified to eliminate parallel operations addressed to the secondary analog console.

Use of a separate digital computer (the IBM TSS/360 computer) allows for the processing of the perturbation data and analysis of the resulting matrices to be done without interfering with use of the hybrid computer system. Calculation of the system matrices could have been done on the Pacer digital computer with the perturbation data stored on the Pacer's associated disk.

The format specified for data output to disk is arbitrary and can be changed if desired. The DISK subroutine of the perturbation program and the data input section of the matrix calculation program should be modified for the desired format. A new checksum error check should be developed to insure correct data transfer from disk.

CONCLUDING REMARKS

The A, B, C, and D state variable matrices used in controls design can be readily determined using the procedure described in this report. Data required to obtain a linear representation of the system to be controlled are systematically gathered using the techniques presented in the text. Disturbances are introduced in the nonlinear system one at a time. The user may determine the level of each disturbance and the sequential order in which each is to be introduced. For each disturbance, corresponding levels of essential system parameters are automatically recorded on floppy disk.

Use of the floppy disk as a storage medium permits the data generation process on the hybrid computer to be separate from the matrix calculation process. This floppy disk data is input to a separate digital program which contains the matrix calculation process. A software checksum is used to insure correct data transmission from the floppy disk. This program also checks the data for proper disturbance levels and correct number of disturbances. Appropriate messages are output if errors are detected, thus allowing data modification, if desired. The partial derivatives forming the elements of the system matrices are approximated by finite difference calculations. The system matrices, appropriately labeled, are printed out by this program.

Use of this procedure on a dynamic system simulated on a hybrid computer generates automatically the data required for linearization of the system and determines the matrices which represent the system.

APPENDIX A

SIMULATION AND PERTURBATION PROGRAMS

Main Program

```
C
C.....LMN = ANALOG COMPUTER MODE INDICATOR
C
      COMMON/Y/MODE/LMN
      COMMON/FLDISC/LF,IXOFF
      LOGICAL*1 SENSW
      CALL OSHV1N (IERROR,681,681)
C.....TAKE ADDRESS OF .FOR AND ADD '213.  SET CONTENTS TO ZERO
      LF = '212
      IXOFF = '223
      CALL SET
      CALL STATIC
   10 CALL OSC (6,IERROR)
      CALL OSC (1,IERROR)
      CALL LOOP
      IF (SENSW(4))  GO TO 30
      CALL STATIC
   20 IF (LMN.EQ.4)  CALL OSOP (IERROR)
      GO TO 10
   30 CALL DATOUT (LMN)
      WRITE (16,61)
   61 FORMAT (/'0% 3('ERESET SSW(A), THEN RESET SSW(D)'/)
      LA  LF
      OCT 3001
      LA  IXOFF
      OCT 3001
   40 IF (SENSW(4))  GO TO 46
      GO TO 20
      END



      SUBROUTINE SET
C
C     ENTER YOUR POTS, DCAS, AND
C     DCFGS TO BE SET AND CHECKED
C
      RETURN
      END
```

```
      SUBROUTINE STATIC
      COMMON/FIELDSC/LP,IXOFF
C
C     ENTER YOUR AMPLIFIERS TO BE KEPT AND CHECKED
C
      WRITE (16,61)
   61 FORMAT (/5X,24HYOU JUST EXECUTED STATIC/)
      LA  LP
      OCT 5001
      LA  IXOFF
      OCT 5001
      RETURN
      END



      SUBROUTINE LOOP
C
C.....LMN = ANALOG COMPUTER MODE INDICATOR
C
      COMMON/XMODE/LMN
      LOGICAL  SENSW
   10 CONTINUE
C
C     ENTER SYSTEM DYNAMIC EQUATIONS INCLUDING
C     BIVARIATE MAPS AND TABLE LOOK-UPS
C
      IF (.NOT.SENSW(1))  GO TO 10
      CALL OSC (2,IERROR)
      CALL GRAM1 (LMN)
      IF (LMN.EQ.4)  CALL OSH (IERROR)
      RETURN
      END



      SUBROUTINE DATOUT (IMODE)
C
C.....N1ST8 = NUMBER OF STATE VARIABLES ON ANALOG CONSOLE 1
C.....  N1IN = NUMBER OF INPUTS ON ANALOG CONSOLE 1
C.....N1OUT = NUMBER OF OUTPUTS ON ANALOG CONSOLE 1
C.....N2ST8 = NUMBER OF STATE VARIABLES ON ANALOG CONSOLE 2
C.....  N2IN = NUMBER OF INPUTS ON ANALOG CONSOLE 2
C.....N2OUT = NUMBER OF OUTPUTS ON ANALOG CONSOLE 2
```

```
C
      COMMON/MEAN/N1ST8,N11OT,N11OTP,AVRG1(56),AMP1SN(25),
     1            N2ST8,N2TOT,N2TOTP,AVRG2(56),AMP2SN(25),IRDG
      COMMON/FLDISC/LF,IXOFF
      DIMENSION  N1DCA(25),N2DCA(25)
      SCALED FRACTION  SGA1N1(25),SGA1N2(25),SV1AMP(25),SV2AMP(25),
     1                 SVALUE
C.....CHECK THAT COMPUTERS WERE IN OPERATE
      IF (IMODE.EQ.4)  GO TO 10
      WRITE (16,61)
   61 FORMAT (/5X,62HCOMPUTERS WERE NOT IN OPERATE WHEN SSW(D) WAS SET.
     1 TRY AGAIN./)
      LA  LF
      OCT 3001
      LA  IXOFF
      OCT 3001
      RETURN
C.....CONSOLE 1 STATE VARIABLES, SYSTEM INPUTS, AND SYSTEM OUTPUTS
   10 READ (6,62) N1ST8,N11N,N10UT
   62 FORMAT (3I5)
      N1TOT = N1ST8+N11N+N10UT
      N1TOTP = N1TOT+N1ST8
      READ (6,63) ((AMP1SN(I),N1DCA(I),SGA1N1(I)),I=1,N1TOT)
   63 FORMAT (F5.1,I3,S7)
C.....CONSOLE 2 STATE VARIABLES, SYSTEM INPUTS, AND SYSTEM OUTPUTS
      READ (6,62) N2ST8,N21N,N20UT
      N2TOT = N2ST8+N21N+N20UT
      N2TOTP = N2TOT+N2ST8
      READ (6,63) ((AMP2SN(I),N2DCA(I),SGA1N2(I)),I=1,N2TOT)
C.....READ STEADY-STATE VALUES OF INTEGRATORS
      CALL OKBADS (SV1AMP,1,N1ST8,IERROR)
C.....CHANGE SIGN TO CORRESPOND TO DCA VALUES
      DO 20 I=1,N1ST8
      SV1AMP(I) = -SV1AMP(I)
   20 CONTINUE
      CALL OKBADS (SV2AMP,35,N2ST8,IERROR)
C.....CHANGE SIGN TO CORRESPOND TO DCA VALUES
      DO 30 I=1,N2ST8
      SV2AMP(I) = -SV2AMP(I)
   30 CONTINUE
C.....SET INTEGRATOR ICS (DCAS) TO STEADY-STATE VALUES
      CALL OSC (0,IERROR)
      CALL OSC (2,IERROR)
      CALL OWDCS (N2DCA,SV2AMP,N2ST8,IERROR)
      CALL OSC (0,IERROR)
      CALL OSC (1,IERROR)
      CALL OWDCS (N1DCA,SV1AMP,N1ST8,IERROR)
C.....PUT COMPUTERS IN IC
      CALL OSC (2,IERROR)
      CALL OSIC (IERROR)
      NUMPTS = 100
```

```
      IRDG = 1
      CALL AVG (NUMPTS)
C.....INPUT VALUES
      DO 40 I=1,N1IN
      J = I+N1ST8
      K = I+2+N1ST8
      SV1AMP(J) = AVKG1(K)+AMP1SN(J)
   40 CONTINUE
      DO 50 I=1,N2IN
      J = I+N2ST8
      K = I+2+N2ST8
      SV2AMP(J) = AVKG2(K)+AMP2SN(J)
   50 CONTINUE
C.....PERTURB CONSOLE 1 VARIABLES
      CALL OSC (6,IERROR)
      CALL OSC (1,IERROR)
      N1VAR = N1ST8+N1IN
      DO 60 I=1,N1VAR
C.....PERTURB VARIABLE POSITIVE
      SVALUE = SV1AMP(I)+SGN1N1(I)*SV1AMP(I)
C.....CHECK IF INPUT VARIABLE
      IF (I.GT.N1ST8)  SVALUE = SGN1N1(I)*SV1AMP(I)
      CALL GWDCS (N1DCH(I),SVALUE,1,IERROR)
      CALL OSDLYK (5.,IERROR)
      CALL LOOP
      CALL AVG (NUMPTS)
      CALL OSC (6,IERROR)
      CALL OSC (1,IERROR)
C.....PERTURB VARIABLE NEGATIVE
      SVALUE = SV1AMP(I)-SGN1N1(I)*SV1AMP(I)
C.....CHECK IF INPUT VARIABLE
      IF (I.GT.N1ST8)  SVALUE = -SGN1N1(I)*SV1AMP(I)
      CALL GWDCS (N1DCH(I),SVALUE,1,IERROR)
      CALL OSDLYK (5.,IERROR)
      CALL LOOP
      CALL AVG (NUMPTS)
      CALL OSC (6,IERROR)
      CALL OSC (1,IERROR)
C.....RESET VARIABLE TO ORIGINAL VALUE
      SVALUE = SV1AMP(I)
C.....CHECK IF INPUT VARIABLE
      IF (I.GT.N1ST8)  SVALUE = .05
      CALL GWDCS (N1DCH(I),SVALUE,1,IERROR)
   60 CONTINUE
C.....PERTURB CONSOLE 2 VARIABLES
      CALL OSC (6,IERROR)
      CALL OSC (2,IERROR)
      N2VAR = N2ST8+N2IN
      DO 70 I=1,N2VAR
C.....PERTURB VARIABLE POSITIVE
      SVALUE = SV2AMP(I)+SGN1N2(I)*SV2AMP(I)
```

```
C.....CHECK IF INPUT VARIABLE
      IF (I.GT.N2STB)  SVALUE = SGN1N2(I)*SV2AMP(I)
      CALL QWDCS (N2DCA(I),SVALUE,1,IERROR)
      CALL QSDLYR (5.,IERROR)
      CALL LOOP
      CALL AVG (NUMPTS)
      CALL QSC (6,IERROR)
      CALL QSC (2,IERROR)
C.....PERTURB VARIABLE NEGATIVE
      SVALUE = SV2AMP(I)-SGN1N2(I)*SV2AMP(I)
C.....CHECK IF INPUT VARIABLE
      IF (I.GT.N2STB)  SVALUE = -SGN1N2(I)*SV2AMP(I)
      CALL QWDCS (N2DCA(I),SVALUE,1,IERROR)
      CALL QSDLYR (5.,IERROR)
      CALL LOOP
      CALL AVG (NUMPTS)
      CALL QSC (6,IERROR)
      CALL QSC (2,IERROR)
C.....RESET VARIABLE TO ORIGINAL VALUE
      SVALUE = SV2AMP(I)
C.....CHECK IF INPUT VARIABLE
      IF (I.GT.N2STB)  SVALUE = .05
      CALL QWDCS (N2DCA(I),SVALUE,1,IERROR)
   70 CONTINUE
      WRITE (16,64)
   64 FORMAT (/5X,'BHCONGRATS.  PERTURBATION JOB COMPLETED./)
      LA  LP
      OCT 5001
      LA  IXOFF
      OCT 5001
      RETURN
      END



      SUBROUTINE SGNP1X
C
C.....N1TOTP = TOTAL NUMBER OF STATE VARIABLES, INPUTS, AND OUTPUTS
C.....ON ANALOG CONSOLE 1
C.....N2TOTP = TOTAL NUMBER OF STATE VARIABLES, INPUTS, AND OUTPUTS
C.....ON ANALOG CONSOLE 2
C
      COMMON/MEAN/N1STB,N1TOT,N1TOTP,AVRG1(56),AMP1SN(25),
     1            N2STB,N2TOT,N2TOTP,AVRG2(56),AMP2SN(25),IRDG
C.....CONSOLE 1 VARIABLES
      DO 30 I=1,N1TOTP
      IF (I.GT.N1STB)  GO TO 10
      AVRG1(I) = AVRG1(I)+AMP1SN(I)
      GO TO 30
```

```
   10 IF (I.GT.(2*N1STB))  GO TO 20
      J = I-N1STB
      AVRG1(I) = -AVRG1(I)*AMP1SN(J)
      GO TO 30
   20 J = I-N1STB
      AVRG1(I) = AVRG1(I)*AMP1SN(J)
   30 CONTINUE
C.....CONSOLE 2 VARIABLES
      DO 60 I=1,N2TOTP
      IF (I.GT.N2STB)  GO TO 40
      AVRG2(I) = AVRG2(I)*AMP2SN(I)
      GO TO 60
   40 IF (J.GT.(2*N2STB))  GO TO 50
      J = I-N2STB
      AVRG2(I) = -AVRG2(I)*AMP2SN(J)
      GO TO 60
   50 J = I-N2STB
      AVRG2(I) = AVRG2(I)*AMP2SN(J)
   60 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE AVG (NUMPTS)
C
C.....NUMPTS = NUMBER OF POINTS TO BE USED PER AVERAGE
C.....N1TOTP = TOTAL NUMBER OF STATE VARIABLES, INPUTS, AND OUTPUTS
C.....ON ANALOG CONSOLE 1
C.....N2TOTP = TOTAL NUMBER OF STATE VARIABLES, INPUTS, AND OUTPUTS
C.....ON ANALOG CONSOLE 2
C
      COMMON/MEAN/N1STB,N1TOT,N1TOTP,AVRG1(50),AMP1SN(25),
     1            N2STB,N2TOT,N2TOTP,AVRG2(50),AMP2SN(25),IRDG
      DIMENSION  SUM1(50),V1AMP(25),SUM2(50),V2AMP(25),TEMP(25)
      DO 10 I=1,N1TOTP
      SUM1(I) = 0.
   10 CONTINUE
      DO 20 I=1,N2TOTP
      SUM2(I) = 0.
   20 CONTINUE
      FLTNO = NUMPTS
C.....GENERATE AVERAGES
      DO 50 K=1,NUMPTS
C.....CONSOLE 2
      CALL  DRRADC (TEMP,33,N2TOTP,IERROR)
      DO 30 I=1,N2TOTP
      SUM2(I) = SUM2(I)+TEMP(I)
```

```
   30 CONTINUE
C.....CONSOLE 1
      CALL UREADK (TEMP,1,N1TOTP,IERROR)
      DO 40 I=1,N1TOTP
      SUM1(I) = SUM1(I)+TEMP(I)
   40 CONTINUE
      CALL USC (6,IERROR)
      CALL USC (1,IERROR)
      CALL LOOP
      CALL USDLYK (5.,IERROR)
   50 CONTINUE
C.....DETERMINE AVERAGES
      DO 60 I=1,N1TOTP
      AVRG1(I) = SUM1(I)/FLINO
      SUM1(I) = 0.
   60 CONTINUE
      DO 70 I=1,N2TOTP
      AVRG2(I) = SUM2(I)/FLINO
      SUM2(I) = 0.
   70 CONTINUE
      CALL SONFIX
      CALL DISK
      RETURN
      END



      SUBROUTINE DISK
C
C.....THE ONLY WAY THAT THE FLOPPY DISK CAN BE WRITTEN ON IS THROUGH
C.....THIS SUBROUTINE
C
      COMMON/MEAN/N1STR,N1TOT,N1TOTP,AVRG1(50),AMP1SN(25),
     1            N2STR,N2TOT,N2TOTP,AVRG2(50),AMP2SN(25),IRDG
      COMMON/FLDISC/LF,IXOFF
      DIMENSION  IA(160),IA10(160),IA1(160)
      WRITE (1,61) IRDG,IRDG
   61 FORMAT (2I5)
      LA  LF
      OCT 3001
      LA  IXOFF
      OCT 3001
      IRDG = IRDG+1
      DO 10 I=1,N1TOTP
      IA(I) = AVRG1(I)*32767.
   10 CONTINUE
      DO 20 I=1,N2TOTP
      J = I+N1TOTP
```

```
      IA(J) = AVRG2(I)*32767.
   20 CONTINUE
      ITOGO = NITOTP+NZTOTP
      INDEX = ITOGO/18
      IF (INDEX.LT.1)  GO TO 66
      DO 50 K=1,INDEX
      ITOT16 = 0
      ITOT1 = 0
      ITSIGN = 1
      DO 30 I=1,18
      J = 18*(K-1)+1
      IF (I.EQ.1)  ISTART = J
      IF (I.EQ.18)  IEND = J
      CALL ADDUP (ITOT16,ITOT1,ITSIGN,IA(J),IA16(J),IA1(J))
   30 CONTINUE
      IF (ITOT16.EQ.0)  GO TO 40
      WRITE (1,62) ((IA16(J),IA1(J)),J=ISTART,IEND),ITOT16,ITOT1
   62 FORMAT (18(I4,I2),I5,I2)
      LA  LF
      OCT 3001
      LA  IXOFF
      OCT 3001
      GO TO 50
   40 ITOT1 = ITSIGN*ITOT1
      WRITE (1,63) ((IA16(J),IA1(J)),J=ISTART,IEND),ITOT1
   63 FORMAT (18(I4,I2),I7)
      LA  LF
      OCT 3001
      LA  IXOFF
      OCT 3001
   50 CONTINUE
   60 NUMBER = ITOGO-18*INDEX
      IF (NUMBER.EQ.0)  RETURN
      ITOT16 = 0
      ITOT1 = 0
      ITSIGN = 1
      DO 70 I=1,NUMBER
      J = 18*INDEX+I
      IF (I.EQ.1)  ISTART = J
      IF (I.EQ.NUMBER)  IEND = J
      CALL ADDUP (ITOT16,ITOT1,ITSIGN,IA(J),IA16(J),IA1(J))
   70 CONTINUE
      WRITE (1,62) ((IA16(J),IA1(J)),J=ISTART,IEND)
      LA  LF
      OCT 3001
      LA  IXOFF
      OCT 3001
      IF (ITOT16.EQ.0)  GO TO 80
      WRITE (1,64)  ITOT16,ITOT1
   64 FORMAT (I5,I2)
      LA  LF
```

```
      OCT 5001
      LA  IXOFF
      OCT 5001
      RETURN
   80 ITOT1 = ITSIGN*ITOT1
      WRITE (1,65) ITOT1
   65 FORMAT (I7)
      LA  IF
      OCT 5001
      LA  IXOFF
      OCT 5001
      RETURN
      END



      SUBROUTINE  ADDUP (ITOT10,ITOT1,ITSIGN,IA,IA10,IA1)
C
C.....ITOT10 = (RUNNING TOTAL)/100
C.....  ITOT1 = ABS(RUNNING TOTAL - 100*ITOT10)
C.....ITSIGN = SIGN OF RUNNING TOTAL
C.....      IA = NEW NUMBER TO BE ADDED TO RUNNING TOTAL
C.....    IA10 = IA/100
C.....     IA1 = ABS(IA - 100*IA10)
C
      IASIGN = ISIGN(1,IA)
      IA10 = IA/100
      IA1 = (ABS(IA-100*IA10)
      IF (IA10.EQ.0)  GO TO 30
      IF ((ITSIGN*IA).LT.0)  GO TO 10
C.....ITOTAL AND IA HAVE THE SAME SIGN
      ITOT10 = ITOT10+IA10
      ITOT1 = ITOT1+IA1
      IF (ITOT1.LT.100)  RETURN
      ITOT1 = ITOT1-100
      ITOT10 = ITOT10+ITSIGN
      RETURN
C.....ITOTAL AND IA HAVE THE DIFFERENT SIGNS
   10 IF (IABS(ITOT10).GT.IABS(IA10))  GO TO 20
C.....ITOTAL AND IA CAN BE ADDED
      ITOTAL = 100*ITOT10+ITSIGN*ITOT1
      ITOTAL = ITOTAL+IA
      ITSIGN = 1
      IF (ITOTAL.LT.0)  ITSIGN = -1
      ITOT10 = ITOTAL/100
      ITOT1 = IABS(ITOTAL-100*ITOT10)
      RETURN
C.....MAGNITUDE OF TOTAL IS GREATER THAN MAGNITUDE OF A
   20 ITOT10 = ITOT10+IA10
```

```
      ITOT1 = ITOT1-IA1
      IF (ITOT1.GE.0)  RETURN
      ITOT1 = ITOT1+100
      ITOT10 = ITOT10-ITSIGN
      RETURN
C     ASSUME A IS ZERO
   30 IA1 = 0
      RETURN
      END
```

MEMORY MAP

C*I*G (600) NUMERICAL NAME	ADR	ZZP	C*I*G (666) ALPHABETICAL NAME	ADR	ZZP	C*I*G (666) REFERENCE NAME	ADR	ZZP
#ZZBOT	00666		ADDUP	06616	*0007	#ZZTOP	00106	
#ZZTOP	00166		AVG	04460	*0034	#MMTOP	21644	
#MMBOT	01000		DATOUT	01465	*0012	#COMBOT	57312	
.MAIN	01000		DISK	05534	*0035	#ZZBOT	00000	
SET	01136	*0063	FLDISC	57777	*0015	#MMBOT	01000	
STATIC	01141	*0064	ABS	11637	*0061	#COTOP	60000	*0016
LOOP	01177	*0066	INLOSS	11657	*0061	.MAIN	01000	
DATOUT	01465	*0012	ISIGN	11666	*0060	FLDISC	57777	*0015
SQNFIX	03342	*0002	LOOP	01177	*0066	MODE	57776	
AVG	04460	*0034	MEAN	57313		USAVIN	07557	*0000
DISK	05534	*0035	MSGFLG	21643		L11	11720	*0001
ADDUP	06616	*0007	OAT	10006	*0002	M11	11725	*0002
UNDCS	07062	*0053	OBASE	10000		SET	01136	*0063
UKAADS	07156	*0026	UCNSTB	07721		STATIC	01141	*0064
UKAADR	07264	*0047	UCUNNO	07730		USC	07553	*0000
UKSWB	07257		UKAM1	07316	*0017	LOOP	01177	*0066
UKSWNB	07265		UKBADK	07204	*0047	SWSW	12161	*0007
UKFWI	07271		UKBADS	07156	*0026	S11	11764	*0010
UKIMI	07300		UKC1	10045		USOP	07512	*0011
UKEMI	07300		UKEM1	07300		DATOUT	01465	*0012
UKAMI	07316	*0017	UKFWI	07271		.FWR	21553	*0013
UKTC1	07320		UKLM1	07300		.FCR	13002	*0014
USDLYR	07333	*0042	UKSWB	07257		UKAM1	07316	*0017
USKUN	07466		UKSWNB	07265		USH	07540	*0020
USKUP	07466		UKTC1	07320		MEAN	57313	
USSTOP	07472		USC	07553	*0000	.FM1	21454	*0021
USKIC	07472		USCLK	07476		.FKE	21413	*0022
USCLR	07476		USDLYR	07333	*0042	.FMK	12571	*0023
USUP	07512	*0011	USKIC	07472		.M11	11732	*0024
USIC	07517	*0053	USKLCN	07733		.FDT	16675	*0025
USUD	07524		USKUP	07466		UKBADS	07156	*0026
USPC	07524		USH	07540	*0020	L44	10362	*0027

```
USKT    07550          USHVIN 07557 *0066 .N44    10612 *0650
USPP    07548          USIC   07517 *0655 .H44    10367 *0651
USST    07554          USOD   07524        0WDCS  07062 *0652
USH     07540 *0026USUP       07512 *0011USIC    07517 *0655
USPS    07545          USPC   07524        AVG    04466 *0654
USSP    07545          USPP   07530        .L22   10621 *0655
USHVIN  07557 *0660USPS       07545        .M22   11300 *0656
UUNSTB  07721          USKT   07550        .C24   10255 *0657
UCONNO  07730          USKUN  07466        .M44   10477 *0040
USELCN  07731          USSP   07545        .H44   10374 *0041
UUPRLM  07746          USST   07554        USDLYK 07555 *0042
.CPU    07751 *0665USSTUP     07472        .S44   10445 *0043
.M111   07752          USTDH  10071        SGNF1X 05542 *0052
USC     07755 *0665UTKNNS     10075 *0064 .M22   10645 *0044
UKC1    10645          UUPRLM 07746        N22    11154 *0045
UKMSE   10655          UWDCS  07062 *0052 .C12   10222 *0046
UM1     10656 *0062SA01       20156 *0105UKKMDK  07204 *0047
USTDA   10071          SC00   12252 *0105 .K22   10717 *0050
UTKNNS  10075 *0664SD00       11657 *0061 .D22   11456 *0051
.C21    10125 *0654SENSW      12161 *0007D1SK    05554 *0053
.C12    10222 *0646SET        01156 *0003 .C21   10125 *0054
.C24    10255 *0657SEVERR     21253 *0104 .D11   12061 *0055
.C42    10557          SGNF1X 05542 *0052 .M11   12017 *0056
.L44    10362 *0027STATIC     01141 *0004MDDUP   06616 *0057
.H14    10367 *0651SVENIN     20665 *0100ISIGN   11661 *0060
.H14    10374 *0041SVENOT     20652 *0102IABS    11657 *0061
.S14    10445 *0643SYSERR     20615 *0101UAT     10056 *0062
.M14    10477 *0040XMODE      57776        UKLMI  07566
.D14    10556          #COBGT 57512        UKTC1  07526
.N44    10612 *0650#COPUP     60000 *0016UKSWB   07257
.L22    10621 *0655#MMBUOT    01066        UKFW1  07271
.H22    10645 *0044#MMTUP     21644        UKFM1  07500
.H22    10717 *0050#ZZBOT     00006        UKSWNB 07265
.S22    11114          #ZZTUP 00106        CPU    05554 *0053
.N22    11154 *0045 .ADAT     21571 *0067USOD    07524
.M22    11300 *0656 .R11      11752 *0024USKT    07550
.D22    11456 *0051 .R22      10717 *0050USST    07554
.NUKM   11662 *0070 .R33      12147        USPS   07545
IRRS    11657 *0061 .R44      10374 *0041USSP    07545
INLOSS  11657 *0061 .CRS      20156 *0103USPC    07524
SD00    11657 *0061 .CKM      17067 *0077USPP    07550
ISIGN   11660 *0060 .CKS      17204 *0072USKUN   07466
.L11    11720 *0661 .CPU      07751 *0665USSTUP  07472
.L55    11720 *0661 .C12      10222 *0046USCLK   07476
.H11    11725 *0662 .C21      10125 *0054USELC   07472
.H55    11725 *0662 .C24      10255 *0657USEUP   07466
.R11    11752 *0024 .C42      10557        UTKNNS 10075 *0064
.S11    11764 *0010 .D11      12061 *0055UKC1    10645
.K11    12017 *0056 .D22      11456 *0051USELCN  07731
.D11    12061 *0055 .D44      10550        UCNSTB 07721
.N11    12142          .FMR   12571 *0...3UBMSE  10655
```

```
 MSS   12147          PRT   21434 +0021 OUPKLM 07746
 MSS   12154          PCR   13002 +0014 ECONNO 07730
SPNSW  12161 +0007  PDT   16675 +0025 M111  07752
 PIO   12252 +0105  PPR   17176 +0076 OSTDA  10071
SCOO   12252 +0105  PIO   12252 +0105 KTMNT 21474 +0066
 PRR   12571 +0025  PLT   17212 +0074 MORO  21571 +0067
 PCR   13002 +0014  POR   17234         C42   16557
 PDT   16675 +0025  PPR   20522 +0071 D44   16536
 HRW   16717 +0075  PRR   21413 +0022 S22   11114
 OEM   17067 +0077  PST   20536         NUKM  11602 +0078
 PPR   17176 +0076  PWR   21553 +0013 NLOSS 11657 +0061
 OKS   17204 +0072  HRW   16717 +0075 SOOO  11657 +0061
 PLT   17212 +0074  H11   11725 +0002 N11   12142
 POR   17234         H22   16645 +0044 LSS   11726 +0001
 OMS   20156 +0105  HSS   11725 +0002 HSS   11725 +0002
SH01   20156 +0105  H44   16567 +0051 RSS   12147
 PPA   20522 +0071  L11   11726 +0001 MSS   12154
 PST   20536         L22   16621 +0050 PPA   20522 +0071
SVSPRR 20615 +0101  LSS   11726 +0001 PIO   12252 +0105
SVPNOT 20652 +0102  L44   16562 +0027 SCOO  12252 +0105
SVPNIN 20663 +0100  MAIN  01000         OKS   17204 +0072
SPVPRR 21233 +0104  M111  07752         WK1   21256 +0073
 WK1   21256 +0073  M11   12017 +0056 PLT   17212 +0074
 PSR   21353 +0013  M22   11300 +0036 HRW   16717 +0075
 PRR   21413 +0022  MSS   12154         PRR   17176 +0076
 PRT   21434 +0021  M44   16477 +0046 OEM   17067 +0077
 KTMNT 21474 +0066  NOKM  11602 +0078 POR   17234
 MORT  21571 +0067  N11   12142         SVPNIN 20663 +0100
MSGFLG 21643         N22   11134 +0045 SVSPRR 20615 +0101
MMIOP  21644         N44   16612 +0053 SVPNOT 20652 +0102
MCOMOT 57312         KTMNT 21474 +0066 OMS   20156 +0105
MPMN   57315         S11   11764 +0010 SH01  20156 +0105
XMODE  57776         S22   11114         PST   20536
PLDISC 57777 +0015  S44   16445 +0043 SPVPRR 21233 +0104
MCOTOP 60000 +0016  WK1   21256 +0073 MSGFLG 21643
DN
```

APPENDIX B

'MATCALC' PROGRAM AND 'READIN' PROCDEF

```
C******MATRIX GENERATION ROUTINE
C
      DIMENSION NAME(30), INDX(61)
      DIMENSION IVAL(10), DATA(61,70)
      DIMENSION NDELT(30), NOBSV(20), NDEIV(20),NCOL(30)
      DIMENSION SF(70),ITEST(30)
      DIMENSION A(20,20),B(20,10),C(20,20),D(20,10)
      DIMENSION HDG(60),ADG(30),NAM2(20),NAM3(20)
C
      DATA DATA/4270*0.0/
      DATA ILAST/0/, INDX/61*0/
      DATA PTEST/.01/, NEROR/0/
      INTEGER VAL,HDG,ADG
      DATA HDG/60*4H    /, ADG/30*4H    /
C
      READ(5,8001) NS, NC, NY
 8001 FORMAT (3I3)
      NWORDS = 2*NS+NC+NY
      NRUNS = 2*(NS+NC)+1
      NTOT = NS+NC
      NSIZE = NS
      NINPUT = NC
C
      READ(5,8002) (NAME(I),I=1,NTOT)
 8002 FORMAT (30A4)
      READ(5,8003)(SF(I),I=1,NWORDS)
 8003 FORMAT (8(10F8.2/))
      READ(5,8002) (NAM2(I),I=1,NS)
      READ(5,8002) (NAM3(I),I=1,NY)
C
C     **********
C     **********
C     **********
C     FLOPPY DISK READ IN AND UNSCALING OF DATA
C
C    **READ IN START
      I=0
    1 I=I+1
      READ(5,5001,END=2,ERR=2) NA, NB
 5001 FORMAT (2I5)
C
C    **DISTURBANCE NUMBER CHECK
      IF (NA .EQ. NB) GO TO 3
      WRITE (6,6100) NA,NB
 6100 FORMAT (5X,'NA VALUE (',I3,') IS NOT EQUAL TO NB VALUE (',I3,')',/,-
     A5X,'ENTER CORRECT DISTURBANCE VALUE (I5) FORMAT')
      READ(7,7101) NA
 7101 FORMAT (I5)
```

```
      GO TO 3
    2 WRITE(6,7002)
 7002 FORMAT (5X,'NO. OF RUNS NOT EQUAL TO EXPECTED NO. OF DISTURBANCES',/,-
     A5X,'ENTER 1 TO STOP, DEFAULT TO CONTINUE'/)
      READ(7,7003) NTEST
 7003 FORMAT (I1)
      IF (NTEST .EQ. 1)  STOP
      GO TO 500
    3 IF (INDX(NA) .EQ. 0) GO TO 4
      WRITE (6,1001) NA ,INDX(NA)
 1001 FORMAT (5X,'INDX(',I2,') PREVIOUSLY SET TO SEQ',I4,/,-
     *       5X,'DEFAULT TO REPLACE, 1 TO STOP'/)
      READ (7,7003) NTEST
      IF (NTEST .EQ. 1) STOP
C
C   **INDX - CONTENTS ARE SEQUENCE NUMBERS
    4 INDX(NA) = I
      ICNT = 1
      J = 1
      NEND = 18
    5 JA = J+17
      IF (LCNT .EQ. 1 .AND. NEND .GT. NWORDS) GO TO 10
      IF (J .GT. NWORDS) GO TO 10
      READ (5,5002) (IVAL(K),K=1,NEND),ICK
 5002 FORMAT (18I6,I7)
      LCNT = LCNT+1
      GO TO 15
   10 JA = NWORDS
      NEND = JA-J+1
      READ (5,5002) (IVAL(K),K=1,NEND)
      READ (5,5003) ICK
 5003 FORMAT (I7)
      LCNT = LCNT+2
C
C   **CHECKSUM CHECK
   15 ISUM = 0
      DO 16 K=1,NEND
   16 ISUM = ISUM+IVAL(K)
      IF (ISUM .NE. ICK) GO TO 50
  100 ILAST = ICK
      WRITE (9,5001) NA,NA
      IF (NEND .NE. 18) GO TO 17
      WRITE (9,5002) (IVAL(K),K=1,NEND),ICK
      GO TO 20
   17 WRITE (9,5002) (IVAL(K),K=1,NEND)
      WRITE (9,5003) ICK
C
C   **UNSCALING OF SAMPLED VARIABLES
   20 K= 0
      DO 25 N=J,JA
      K = K+1
      YI = IVAL(K)
```

```
      YS = YI/32768.
   25 DATA(I,N) = YS*SF(N)
      J = JA+1
      IF (J .LE. NWORDS) GO TO 5
      IF (I .LT. NRUNS) GO TO 1
      WRITE (6,6001)
 6001 FORMAT (5X,'MATRIX DATA READ IN AND UNSCALING COMPLETE'/,-
     A5X,'DATASET ON UNIT 9 CONTAINS (CORRECTED IF NEC) FLOPPY DISK DATA')
      GO TO 500
C
C    **CHECKSUM ERROR ROUTINE
   50 WRITE (6,6101) NA, I, LCNT
 6101 FORMAT (' CHECKSUM ERROR',3X,' RUN ',I3,' SEQ ',I3, '  LINE ',I4)
      WRITE (6,6102) ILAST, ICK, ISUM
 6102 FORMAT (' CHECKSUMS  : PREVIOUS = ',I7, '  READ = ',I7,'  CALC = ',I7)
      WRITE (6,6103) (IVAL(K),K=1,NEND)
 6103 FORMAT ( ' THE VALUES ARE ',/,1X,1816,/)
  200 WRITE (6,6104)
 6104 FORMAT (' ENTER CORRECTED LINE')
      IF (NEND .EQ. 18) GO TO 6010
      READ (7,5002) (IVAL(K),K=1,NEND)
      READ (7,5003) ICK
      GO TO 6020
 6010 READ (7,5002) (IVAL(K),K=1,NEND),ICK
 6020 ISUM = 0
      DO 55 K=1,NEND
   55 ISUM = ISUM+IVAL(K)
      IF (ISUM .EQ. ICK) GO TO 100
      WRITE (6,6105) ISUM ,ICK
 6105 FORMAT (' CHECKSUMS DIFFER,   CALC = ',I7,'    READ = ',I7)
      WRITE (6,6106)
 6106 FORMAT (' ENTER 1 TO RE-ENTER , 2 TO STOP,DEFAULT TO CONTINUE')
      READ (7,7003) NTEST
      NTEST = NTEST+1
      GO TO (100,200,300),NTEST
  300 STOP
C
C     **********
  500 CONTINUE
C
C     IF DESIRE A LISTING OF DATA MATRIX WHICH NOW CONTAINS UNSCALED DISTURBANCE DATA,
C     INSERT WRITE STATEMENTS HERE
C
C
C     **********
C     **  ******
C     **********
C     INITIALIZATION FOR DISTURBANCE VERIFICATION
C
      NBRUN = INDX(1)
      IF (NBRUN .NE. 0) GO TO 544
      WRITE (6,6002)
```

```
 6002 FORMAT (5X,'NO BASE DATA,TERMINATE')
      STOP
  544 READ(8,8005) (NDELT(K),K=1,NTOT)
      READ(8,8005) (NDRIV(K),K=1,NY)
      READ(8,8005) (NOBSV(K),K=1,NY)
      READ (8,8005) (NCOL(K),K=1,NTOT)
 8005 FORMAT(30I3)
C
      DO 510 N=1,NTOT
      ND = NDELT(N)
  510 ATEST(N) = SF(ND)/1000.
C     *********
C     *********
C     DISTURBANCE CHECK AND VERIFICATION
C
      DO 700 N=1,NTOT
      NN=N-NSIZE
      NS3 = NAME(N)
      DO 600 M=1,2
      NC=0
      NRI = 2*N+M-1
      NI = INDX(NRI)
      IF (NI .EQ. 0) NI = NBRUN
      DO 400 I=1,NTOT
      ND = NDELT(I)
      IF (ABS(DATA(NBRUN,ND)) .LT. ATEST(I)) GO TO 399
      TEST = (DATA(NI,ND)-DATA(NBRUN,ND))/DATA(NBRUN,ND)
      IF (ABS(TEST) .LT. PTEST) GO TO 400
  332 IF (I .NE. N) GO TO 333
      NC = I
      GO TO 400
  333 NS1 = NAME(I)
      NEROR = NEROR+1
      WRITE (6,201) NI,NRI,NS1,NS3,DATA(NI,ND),DATA(NBRUN,ND)
  201 FORMAT (1H ,'SEQUENCE ',I2,' INDX ',I2,2H, ,A4,' IS PERTURBED IN ADDITION TO ',A4,/,-
     AX,'VALUE =',G13.5,' BASE VALUE = ',G13.5)
      WRITE (6,203) NI
  203 FORMAT (' ENTER 1 TO USE BASE INSTEAD OF SEQ ',I2,' OTHERWISE DEFAULT')
      READ (7,7003) NTEST
      IF (NTEST .NE. 1) GO TO 400
      INDX(NRI) = 0
      WRITE(6,204) NI
  204 FORMAT (5X,'BASE IS BEING USED FOR SEQ ',I2,/)
      GO TO 400
  399 TEST = DATA(NI,ND)-DATA(NBRUN,ND)
      IF (ABS(TEST) .GT. 2*ATEST(I)) GO TO  332
  400 CONTINUE
      IF (NC .EQ. N) GO TO 600
      IF (INDX(NRI) .EQ. 0) GO TO 590
      NEROR = NEROR+1
      ND = NDELT(N)
      WRITE (6,202) NI,NRI,NS3,DATA(NI,ND),DATA(NBRUN,ND)
```

```
  202 FORMAT (1H ,'SEQUENCE ',I2,', INDX ',I2,', ',A4,' DISTURBANCE LOW OR ZERO',/,-
     A8X,'VALUE = ',G13.5,' BASE VALUE = ',G13.5)
      GO TO 600
  590 WRITE (6,209) NRI,NS3
  209 FORMAT (' INDX ',I2,', NO DISTURBANCE FOR ',A4,/,-
     A' USING BASE RUN')
  600 CONTINUE
  700 CONTINUE
C
C     **********
C     **********
C     **********
C     MATRIX CALCULATION
C
      J=-1
      DO 800 N=1,NTOT
      NCL = NCOL(N)
      NN = N-NSIZE
      NS3 = NAME(NCL)
      NWORD = NDELT(NCL)
      NTWO = 2*NCL
      NROW1 = INDX(NTWO)
      NROW2 = INDX(NTWO+1)
      IF (NROW1 .EQ. 0)NROW1 = NBRUN
      IF (NROW2 .EQ. 0) NROW2 = NBRUN
  607 DNOMN = DATA(NROW1,NWORD)-DATA(NROW2,NWORD)
      IF (ABS(DNOMN) .GT. .001) GO TO 609
      IF (NROW1 .EQ. NBRUN) GO TO 608
      NI = NROW1
      WRITE (6,205) NS3,NI
  205 FORMAT (1H ,A4,' DISTURBANCE IDENTICAL, SETTING SEQUENCE ',I2,' TO BASE')
      NROW1 = NBRUN
      GO TO 607
  608 WRITE (6,206) NS3
  206 FORMAT(' NO ',A4,' DISTURBANCES, MATRIX COLUMN ZEROED')
      DNOMN = 1.0E50
C  ***CALCULATION OF A & B MATRIX COLUMNS
  609 DO 620 K=1,NSIZE
      NDUM = NDRIV(K)
      IF (N .GT. NSIZE) GO TO 610
      A(K,N) = (DATA(NROW1,NDUM)-DATA(NROW2,NDUM))/DNOMN
      GO TO 620
  610 B(K,NN) = (DATA(NROW1,NDUM)-DATA(NROW2,NDUM))/DNOMN
  620 CONTINUE
C  ***CALCUALATION OF C & D MATRIX COLUMNS
      DO 630 K=1,NSIZE
      NDUM = NOBSV(K)
      IF (N .GT. NSIZE) GO TO 625
      C(K,N) = (DATA(NROW1,NDUM)-DATA(NROW2,NDUM))/DNOMN
      GO TO 630
  625 D(K,NN) = (DATA(NROW1,NDUM)-DATA(NROW2,NDUM))/DNOMN
```

```
  630 CONTINUE
      J=J+3
      VAR = NCOL(N)
      IF(N .GT. NS+1) GO TO 710
      IF(N .GT. NS) GO TO 705
      HDG(J) = NAME(VAR)
      GO TO 800
  705 J= 2
  710 ADG(J) = NAME(VAR)
  800 CONTINUE
C
C     **********
C
C     PRINTOUT OF MATRICES
C     10 COLS MAX PER LINE
      NC = NINPUT
      WRITE (6,6900)
 6900 FORMAT (5X,'THE A MATRIX'/)
      IF (NS .LE. 10) GO TO 6950
      NCOL1=NS/2
      NCOL2=NCOL1+1
      WRITE (6,6960) (HDG(L),L=1,30)
 6960 FORMAT(' ',9X,30A4)
      DO 6906 I=1,NS
 6906 WRITE (6,6901) NAM2(I),(A(I,J),J=1,NCOL1)
      WRITE (6,6960) (HDG(L),L=31,60)
      DO 6907 I=1,NS
 6907 WRITE (6,6901) NAM2(I),(A(I,J),J=NCOL2,NS)
      GO TO 6955
 6950 WRITE (6,6960) (HDG(L),L=1,30)
      DO 6908 I=1,NS
 6908 WRITE (6,6901) NAM2(I),(A(I,J),J=1,NS)
 6901 FORMAT (' ',A4,5X,10G12.4/)
 6955 WRITE (6,6902)
 6902 FORMAT (5X,'THE B MATRIX'/)
      WRITE (6,6960) (ADG(L),L=1,30)
      DO 6911 I=1,NS
 6911 WRITE (6,6901) NAM2(I),(B(I,J),J=1,NC)
      WRITE(6,6903)
 6903 FORMAT(5X,'THE C MATRIX'/)
      IF (NS .LE. 10) GO TO 6970
      WRITE (6,6960) (HDG(L),L=1,30)
      DO 6921 I=1,NY
 6921 WRITE (6,6901) NAM3(I),(C(I,J),J=1,NCOL1)
      WRITE (6,6960) (HDG(L),L=31,60)
      DO 6922 I=1,NY
 6922 WRITE (6,6901) NAM3(I),(C(I,J),J=NCOL2,NS)
      GO TO 6975
 6970 WRITE (6,6960) (HDG(L),L=1,30)
      DO 6923 I=1,NY
 6923 WRITE (6,6901) NAM3(I),(C(I,J),J=1,NS)
 6975 WRITE (6,6904)
```

```
 6904 FORMAT(5X,'THE D MATRIX'/)
      WRITE (6,6960) (ADG(I),I=1,30)
      DO 6931 I=1,NY
 6931 WRITE(6,6901) NAM3(I),(D(I,J),J=1,NC)
C
C     IF DESIRE A PERMANENT DATASET WITH A,B,C,D MATRICES
C     INSERT WRITE STMTS HERE WITH DESIRED FORMAT
C
C
C   **ERROR MESSAGE NUMBER PRINTOUT
      IF (NEROR .NE. 0) WRITE(6,711) NEROR
  711 FORMAT (///,'  *** AT LEAST ',I3,' ERRORS ***')
      STOP
      END
```

```
PROCDEF READIN

PROCDEF READIN
PARAM DTRK
IF ('DTRK'=''); GO TO 'END'
DISPLAY 'ENTER DATA - FINISH WITH _END'
ERASE DTRK
EDIT DTRK
DEFAULT SYSLNX=5
REDIT DTRK
VSFILE DTRK;0
DEFAULT SYSLNX=0
DISPLAY 'VS-DATASET COMPLETED - MODIFY IF NEC'
'END'
```

APPENDIX C

'QINFO' dataset

Format:

line 1	NS, NC, NY	3I3
	NAME array	30A4
	SF array	8(10F8.2)
	NAM2 array	30A4
	NAM3 array	30A4
	NDELT array	30I3
	NDRIV array	30I3
	NOBSV array	30I3
last line	NCOL array	30I3

Description:

NS : number of states
NC : number of inputs
NY : number of outputs
NAME: names of the states and inputs in the order perturbed for generation of hybrid data.
length of each name - 4 alphanumeric characters
Each position of the NAME array contains the name of a state or an input. Since each state and each input is perturbed in both a plus and minus direction, each position of the NAME array corresponds to two readings and hence, two disturbance numbers. The correspondence between a position in the NAME array and two disturbance numbers is as follows:

N: position in NAME array
(e.g., NAME(N) = ST1 name of state 1)
disturbance number of plus perturbation 2*N
disturbance number of minus perturbation 2*N+1

NCOL: determines columnar structure of the matrices
The integer values represent the positions of the NAME array in the order desired for the columns of matrices.
(e.g., NAME array ST1, ST2, IN1, IN2, ST3, ST4, IN3, IN4
position 1 2 3 4 5 6 7 8
Want columns 1, 2, 3, and 4 of A and C matrices to be with respect to ST1, ST2, ST3, ST4 and want columns 1, 2, 3, and 4 of B and D matrice to be with respect to IN1, IN2, IN3, IN4.

Then NCOL array is 1, 2, 5, 6, 3, 4, 7, 8)

The data for the arrays SF, NDELT, NDRIV, and NOBSV depend on the order of the values in the rows of the DATA matrix. Let 'ROW' mean the general form of a row in the DATA matrix.

SF scale factors corresponding to values in 'ROW'.
the real x values in 1/x scale factor representation.

NDELT: positions in 'ROW' occupied by the values of the states and inputs.

NDRIV: positions in 'ROW' occupied by the values of the state derivatives ordered as the rows of A and B matrices are desired.

NOBSV: positions in 'ROW' occupied by the values of the outputs ordered as rows of C and D matrices are desired.

NAM2: names of the state derivatives in order specified in NDRIV.
length per name - 4 alphanumeric characters.

NAM3: names of the outputs in order specified in NOBSV.
length per name - 4 alphanumeric characters.

APPENDIX D

DATA PROCESSING PROCEDURE

Note: This procedure is specialized for use with the NASA-Lewis Research Center EAI Pacer 100 computer system.

TEKTRONIX 4010 TERMINAL INITIALIZATION

1. turn terminal ON (power light will come on).
2. turn 360 switch to the ON position.
3. turn SEL switch to the OFF position.
4. set BAUD switch for 1200.
5. set NORM/TP-WR switch for NORM.
6. set LOCAL/LINE switch for LINE.
7. on the phone modem, momentairly depress the CONNECT button.
   (the CONNECT light will come on when line is connected).
8. hit 'BREAK' key on terminal.
9. 'logon' to 360 as normally done.
   (check TERMINAL ID; it should be posted).

TEKTRONIX 4922 FLOPPY DISK INITIALIZATION

1. turn POWER switch ON.
2. depress RESET button.
3. turn PROMPT MODE switch OFF.
4. select proper disk.
   a. check disk notebook.
   b. make sure disk moves freely in envelope.
5. set DISK switch for disk being used.
   a. A if disk 1.
   b. B if disk 2.
6. set TERMINAL switch (switch #1) for terminal being used.
   a. LEFT if left terminal.
   b. RIGHT if right terminal.
7. set TRANSMISSION switch (switch #2) to the LEFT.
   (LEFT setting - data transmission stops at 'XOFF' character).
8. set R-W switch to MANUAL READ.

DATA TRANSFER FROM FLOPPY DISK

1. depress RESET button on disk.
2. set beginning track address in TRACK ADDRESS.
3. depress LA button on disk.
4. type at terminal

   READIN 'DTRK' (CR).

   when the procdef starts, the message:

   'ENTER DATA - FINISH WITH - END'

   will print at the terminal. Data transfer will be automatic; the user can determine this by the periodic "noise" the disk will make during this transfer.
5. type at terminal:

   _END (CR)

   when the end of the data is reached.
   This will end the EDIT dataset created by procdef READIN.
   It may be necessary to type _END (CR) more than once before the 360 accepts it. These unaccepted lines are entered into the dataset 'DTRK' and must be deleted through REDIT before using 'DTRK' as input to the MATCALC program.

'MATCALC' PROGRAM EXECUTION

After creation of datasets 'DTRK' and "QINFO', the program can be executed by typing at the terminal:

```
DDEF   FT05F001,VS, 'DTRK'
DDEF   FT08F001,VS, 'QINFO'
DDEF   FT09F001,VS, 'QOUTPUT'
MATCALC
```

NOTES:

1. When the message 'MATRIX DATA READ IN' etc. is printed at the terminal, depress RESET button on disk.

2. Final printout at terminal will be the A, B, C, and D matrices

3. If errors are detected, appropriate messages will be printed out, allowing the user opportunity to make corrections. The procedure for checksum error correction follows.

CHECKSUM ERROR PROCEDURE

If the read and calculated values of the checksum for a line of floppy disk data are not equal, a checksum error has occurred. An error message will print at the terminal, allowing the data line to be re-entered as follows:

1. set LOCAL/LINE switch to LOCAL on terminal.
2. depress RESET button on disk.
3. set beginning track address in TRACK ADDRESS.
4. depress LA button.
5. at the terminal,
   the 'CTRLQ' key will input one line of floppy disk data to the terminal CRT screen.
   a. Lines of data should be inputted until the proper disturbance reading data is reached. This is done by comparing the RUN number outputted in the error message to the values in the line containing the disturbance number for each reading.
   b. Lines of data from that reading should then be inputted, noting the checksum for each line. When the checksum equal to the PREVIOUS checksum (printed in the error message) is reached, the LOCAL/LINE switch should be set to LINE.
   c. The next data line should be inputed by the 'CTRLQ' key. This line will be accepted by the 360 as the re-entered data line. If the checksum of this data line is on the next line, that line will automatically input to the 360.

TABLE I. - PERTURBATION CONTROL CARDS FOR 4-STATE, 4-INPUT, 4-OUTPUT EXAMPLE

```
2  2  2
 1.  45      .02
 1.  75      .02
-1.  41      .02
-1.  71      .02
-1.
-1.
2  2  2
 1.  75      .02
 1.  45      .02
-1.  71      .02
-1.  41      .02
-1.
-1.
```

TABLE II. - PERTURBATION ORDER OF STATE VARIABLES AND SYSTEM INPUTS FOR 4-STATE, 4-INPUT, 4-OUTPUT EXAMPLE

Parameter	Console	ADC Channel
ST1	Primary	1
ST2	Primary	2
IN1	Primary	5
IN2	Primary	6
ST3	Secondary	33
ST4	Secondary	34
IN3	Secondary	37
IN4	Secondary	38

Note: The perturbation order of state variables and system inputs per console is arbitrary.

TABLE III. - READING NUMBERS FOR 4-STATE,

4-INPUT, 4-OUTPUT EXAMPLE

Rdg. #	(= disturbance #)
1	BASE RUN
2	ST1 +
3	ST1 -
4	ST2 +
5	ST2 -
6	IN1 +
7	IN1 -
8	IN2 +
9	IN2 -
10	ST3 -
11	ST3 -
12	ST4 +
13	ST4 -
14	IN3 +
15	IN3 -
16	IN4 +
17	IN4 -

Order of values per reading:
ST1, ST2, DRV1, DRV2, IN1, IN2, OUT1, OUT2, ST3, ST4, DRV3, DRV4, IN3, IN4, OUT3, OUT4

Note - STJ represents state variable x_J where J = 1 to 4

INJ represents input variable u_J where J = 1 to 4

OUTJ represents output variable y_J where J = 1 to 4

DRVJ represents state derivative $\dot{x}_J$ where J = 1 to 4

TABLE IV. - FLOPPY DISK DATA FOR 4-STATE, 4-INPUT, 4-OUTPUT EXAMPLE

```
   1    1
16392 24500      0      0 16381 24574 16384 24581 24606 16419      0      0 24576 16382 24598 16409
245898
   2    2
16719 24595   -135      0 16383 24575 15709 24581 24603 16416      0      0 24576 16383 24595 16410
246210
   3    3
15976 24592    104      0 16383 24575 16064 24584 24603 16416      0      0 24576 16383 24595 16410
245561
   4    4
16388 25091      0   -508 16383 24575 16385 25080 24603 16416      0      0 24576 16383 24595 16410
246377
   5    5
16388 24112      0    403 16383 24575 16385 24099 24602 16416      0      0 24576 16383 24595 16410
245387
   6    6
16388 24596    104      0 16698 24575 16385 24581 24602 16416      0      0 24576 16383 24595 16410
246503
   7    7
16389 24597   -135      0 16054 24575 16385 24581 24603 16416      0      0 24576 16383 24595 16410
245229
   8    8
16388 24596      0    463 16383 24585 16385 24581 24602 16416      0      0 24576 16383 24595 16410
246843
   9    9
16389 24597      0   -507 16383 24084 16385 24581 24603 16416      0      0 24576 16383 24595 16410
246895
  10   10
16391 24599      0      0 16381 24575 16384 24582 25098 16418   -513      0 24576 16382 25089 16410
246872
  11   11
16388 24597      0      0 16383 24575 16385 24581 24120 16415    459      0 24576 16383 24112 16410
245384
  12   12
16389 24597      0      0 16383 24575 16385 24581 24603 16748      0   -363 24577 16383 24595 16743
246196
  13   13
16388 24596      0      0 16383 24575 16385 24581 24603 16101      0    279 24577 16383 24596 16095
245842
  14   14
16388 24597      0      0 16383 24575 16385 24581 24603 16416    458      0 25069 16383 24595 16410
246943
  15   15
16388 24597      0      0 16383 24575 16385 24580 24603 16416   -511      0 24088 16383 24595 16410
244892
  16   16
16389 24597      0      0 16383 24575 16385 24581 24603 16416      0    276 24577 16706 24596 16410
246494
  17   17
16388 24597      0      0 16383 24575 16385 24581 24603 16416      0   -362 24577 16059 24596 16410
245208
```

TABLE V. - 'INDX' ARRAY AND 'DATA' MATRIX FOR 4-STATE, 4-INPUT, 4-OUTPUT EXAMPLE

INDX ARRAY

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17

DATA MATRIX

Row	Column 1	2	3	4	5	6	7	8
1	1.00049	1.87683	0.000000	0.000000	0.999817	1.87485	1.00000	1.87538
2	1.02045	1.87645	-.204468E-010	0.000000	0.999939	1.87492	1.01984	1.87538
3	0.981201	1.87622	0.185547E-010	0.000000	0.999939	1.87492	0.980469	1.87561
4	1.00024	1.91429	0.000000	-.387573E-010	0.999939	1.87492	1.00006	1.91345
5	1.00024	1.83960	0.000000	0.353241E-010	0.999939	1.87492	1.00006	1.83861
6	1.00024	1.87653	0.185547E-010	0.000000	1.01917	1.87492	1.00006	1.87538
7	1.00031	1.87660	-.204468E-010	0.000000	0.979858	1.87492	1.00006	1.87538
8	1.00024	1.87653	0.000000	0.353241E-010	0.999939	1.91231	1.00006	1.87538
9	1.00031	1.87660	0.000000	-.386810E-010	0.999939	1.83746	1.00006	1.87538
10	1.00043	1.87675	0.000000	0.000000	0.999817	1.87492	1.00000	1.87546
11	1.00024	1.87660	0.000000	0.000000	0.999939	1.87492	1.00006	1.87538
12	1.00031	1.87660	0.000000	0.000000	0.999939	1.87492	1.00006	1.87538
13	1.00024	1.87653	0.000000	0.000000	0.999939	1.87492	1.00006	1.87538
14	1.00024	1.87660	0.000000	0.000000	0.999939	1.87492	1.00006	1.87538
15	1.00024	1.87660	0.000000	0.000000	0.999939	1.87492	1.00006	1.87531
16	1.00031	1.87660	0.000000	0.000000	0.999939	1.87492	1.00006	1.87538
17	1.00024	1.87660	0.000000	0.000000	0.999939	1.87492	1.00006	1.87538

Row	9	10	11	12	13	14	15	16
1	3.75458	2.00427	0.000000	0.000000	3.75000	1.99976	3.75275	2.00305
2	3.75412	2.00391	0.000000	0.000000	3.75000	1.99988	3.75290	2.00317
3	3.75412	2.00391	0.000000	0.000000	3.75000	1.99988	3.75290	2.00317
4	3.75412	2.00391	0.000000	0.000000	3.75000	1.99988	3.75290	2.00317
5	3.75397	2.00391	0.000000	0.000000	3.75000	1.99988	3.75290	2.00317
6	3.75397	2.00391	0.000000	0.000000	3.75000	1.99988	3.75290	2.00317
7	3.75412	2.00391	0.000000	0.000000	3.75000	1.99988	3.75290	2.00317
8	3.75397	2.00391	0.000000	0.000000	3.75000	1.99988	3.75290	2.00317
9	3.75412	2.00391	0.000000	0.000000	3.75000	1.99988	3.75290	2.00317
10	3.83965	2.00415	-.782776E-010	0.000000	3.75000	1.99976	3.82828	2.00317
11	3.68042	2.00378	0.700378E-010	0.000000	3.75000	1.99988	3.67920	2.00317
12	3.75412	2.04443	0.000000	-.443115E-01	3.75015	1.99988	3.75290	2.04382
13	3.75412	1.96545	0.000000	0.340576E-01	3.75015	1.99988	3.75305	1.96472
14	3.75412	2.00391	0.698853E-010	0.000000	3.82523	1.99988	3.75290	2.00317
15	3.75412	2.00391	-.779724E-010	0.000000	3.67554	1.99988	3.75290	2.00317
16	3.75412	2.00391	0.000000	0.336914E-01	3.75015	2.03931	3.75305	2.00317
17	3.75412	2.00391	0.000000	-.441895E-01	3.75015	1.96033	3.75305	2.00317

TALLE VI. - 'QINFO' DATASET FOR 4-STATE, 4-INPUT, 4-OUTPUT EXAMPLE

```
  4   4   4
ST1 ST2 IN1 IN2 ST3 ST4 IN3 IN4
      2.      2.5      2.      2.5      2.      2.5      2.      2.5      5.      4.
      5.      4.       5.      4.       5.      4.
DRV1DRV2DRV3DRV4
OUT1OUT2OUT3OUT4
  1   2   5   6   9  10  13  14
  3   4  11  12
  7   8  15  16
  1   2   5   6   3   4   7   8
```

Note: Appendix C gives the required format specifications for 'QINFO' dataset and defines the arrays in the MATCALC program which are initialized by this data.

TABLE VII. - A, B, C, D MATRIX RESULTS FOR 4-STATE, 4-INPUT, 4-OUTPUT EXAMPLE

THE A MATRIX

	ST1	ST2	ST3	ST4
DRV1	-0.9938	0.0000	0.0000	0.0000
DRV2	0.0000	-0.9918	0.0000	0.0000
DRV3	0.0000	0.0000	-0.9939	0.0000
DRV4	0.0000	0.0000	0.0000	-0.9923

THE B MATRIX

	IN1	IN2	IN3	IN4
DRV1	0.9922	0.0000	0.0000	0.0000
DRV2	0.0000	0.9888	0.0000	0.0000
DRV3	0.0000	0.0000	0.9878	0.0000
DRV4	0.0000	0.0000	0.0000	0.9861

THE C MATRIX

	ST1	ST2	ST3	ST4
OUT1	1.003	0.0000	-0.4090E-03	0.0000
OUT2	-0.5932E-02	1.002	0.5112E-03	0.0000
OUT3	0.0000	0.0000	0.9990	-0.1932E-02
OUT4	0.0000	0.0000	0.0000	1.002

THE D MATRIX

	IN1	IN2	IN3	IN4
OUT1	0.0000	0.0000	0.0000	0.0000
OUT2	0.0000	0.0000	0.5097E-03	0.0000
OUT3	0.0000	0.0000	0.0000	0.0000
OUT4	0.0000	0.0000	0.0000	0.0000

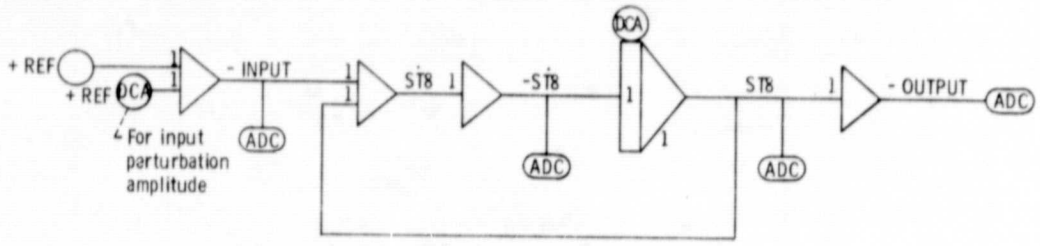

Figure 1. - General analog circuit based on $ST8 = \int (INPUT - ST8)dt$.

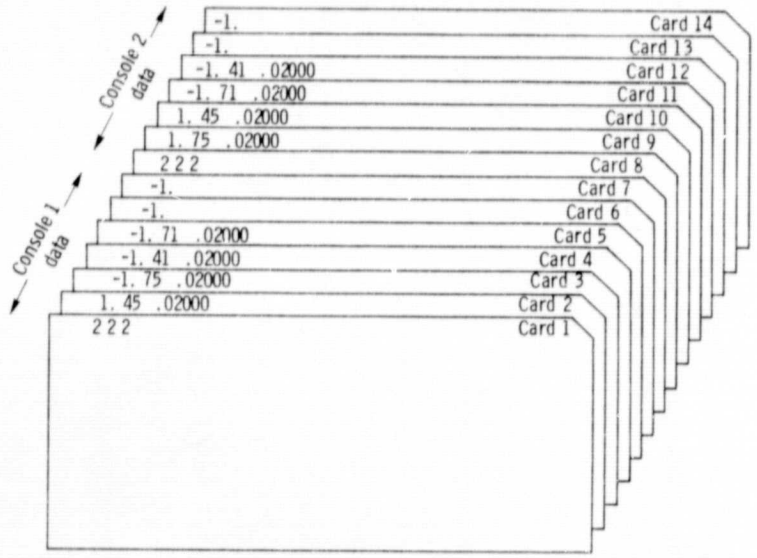

Figure 2. - Data deck setup for example.